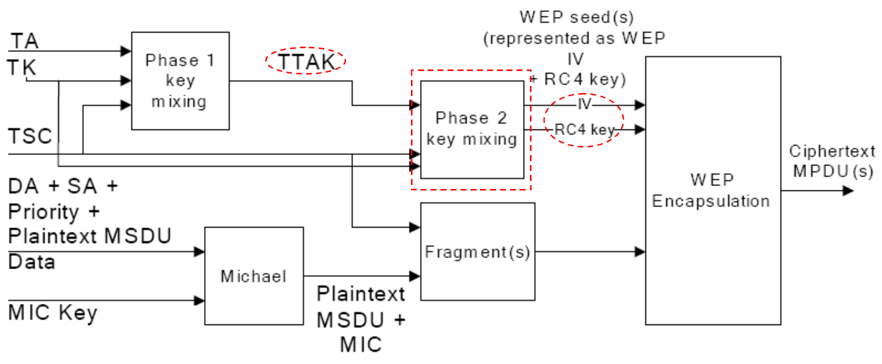
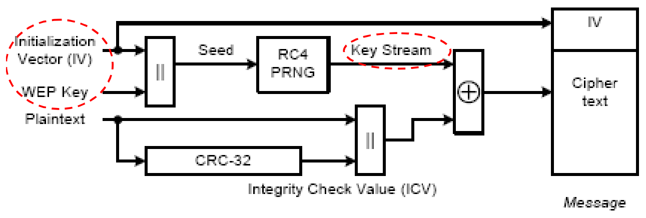
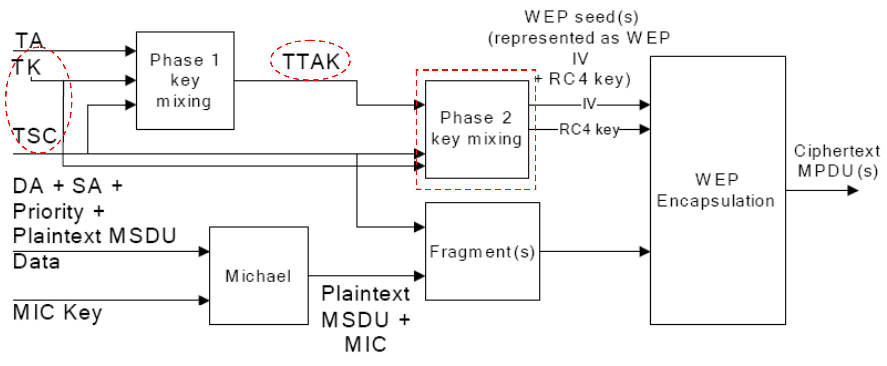
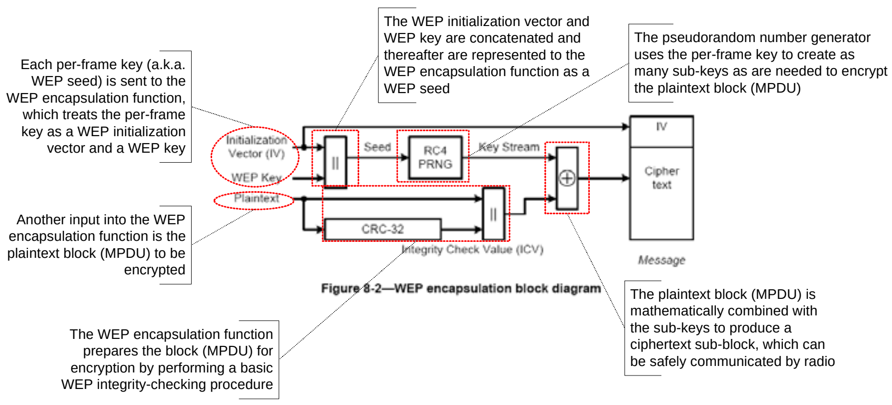
EXHIBIT C

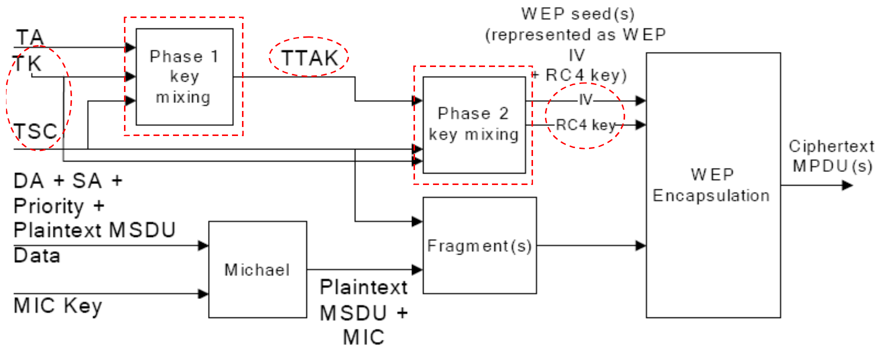
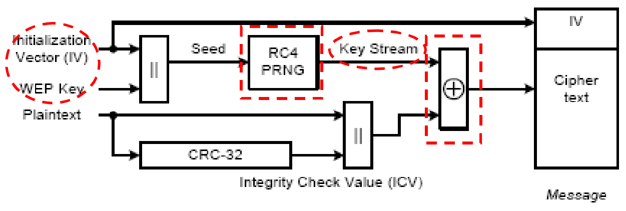
Attachment C - Exemplary Claim Chart Comparing U.S. Patent No. 6,259,789 to Temporal Key Integrity Protocol (TKIP) Technology

U.S. Patent No. 6,259,789	Temporal Key Integrity Protocol (TKIP)
1. A computer implemented method for encrypting data comprising the steps of:	TKIP provides a computer implemented method for encrypting data. *See, e.g.*, IEEE Std 802.11-2007, at PAONE0015307-16538 ("IEEE 802.11 Standard") (PAONE bates-numbered excerpts included as Attachment B). TKIP was designed to provide data encryption superior to WEP, an earlier encryption standard, without requiring substantially more computing power than WEP required. *See, e.g.*, IEEE 802.11 Standard at PAONE0015392 ("The TKIP is a cipher suite enhancing the WEP protocol on pre-RSNA hardware."). TKIP is and has been implemented on a computer. *See, e.g.*, *id.* at PAONE0016461-485 (computer source code implementation of TKIP functions).
creating at least one object key in a block cipher, the at least one object key comprising data and methods that operate on said data;	TKIP creates an object key in a block cipher that comprises data and methods that operate on the data. TKIP is a block cipher because it operates by encrypting or decrypting blocks of data. For example, TKIP operates on blocks called medium access control protocol data units (MPDUs). *See, e.g.*, *id.* at PAONE0015364, PAONE0015520. TKIP creates an object key comprising data and methods that operate on said data. The initial state of the object key is the TKIP-mixed transmit address and key (TTAK). The initial state of the data includes the "keying material in[] the 80-bit TTAK." *Id.* at PAONE0015531. The TKIP object key also comprises methods that operate on said data, including the phase 2 key-mixing function. The phase 2 key-mixing function operates on the data in the TTAK to produce subsequent modified versions of the object key, described as per-frame keys, for each block of input plaintext data. *See, e.g.*, *id.* at PAONE0015520-521, PAONE0015529, PAONE0015531. Thus, in subsequent iterations of the process, the state of the object key data is the per-frame key, also called a WEP seed. The 128-bit per-frame key consists of a 24-bit WEP initialization vector (IV) and a 104-bit RC4 key. *See, e.g.*, *id.* at PAONE0015513, PAONE0015520.

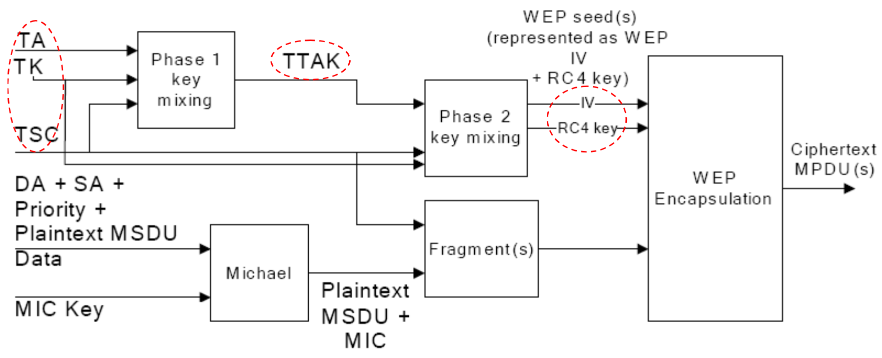
	 TA TK TSC Phase 1 key mixing TTAK WEP seed(s) (represented as WEP IV + RC4 key) Phase 2 key mixing IV RC4 key WEP Encapsulation Ciphertext MPDU(s) DA + SA + Priority + Plaintext MSDU Data MIC Key Michael Plaintext MSDU + MIC Fragment(s) **Figure 8-4—TKIP encapsulation block diagram** *Id.* at PAONE0015520. For example, as illustrated above in Figure 8-4 of the IEEE 802.11 Standard, the block cipher encryption process inputs the transmitter address value (TA), the temporal key value (TK), and the TKIP sequence counter value (TSC) to the phase 1 key mixing function to create a TKIP-mixed transmit address and key value (TTAK). The block cipher encryption process also inputs the TKIP-mixed transmit address and key value (TTAK), the temporal key value (TK), and the TKIP sequence counter value (TSC) to the phase 2 key mixing function to generate the WEP seed. *See also, e.g., id.* at PAONE0015519-533.
creating a key schedule based upon the at least one object key;	TKIP creates a key schedule based on the object key. For example, TKIP uses the per-frame key to create a key schedule for each block to be encrypted. The per-frame key's 128-bit key value is expanded, using the RC4 pseudorandom number generator, to the length necessary to encrypt an entire block of data. *See, e.g., id.* at PAONE0015513-514, PAONE0015531. An MPDU may be as long as 2,346 bytes, and TKIP requires a key schedule of equal length. *See, e.g., id.* at PAONE0015513. Accordingly, TKIP expands the per-frame key's 128 bits of data to create a key schedule as long as 18,768 bits (2,346 bytes).  Initialization Vector (IV) WEP Key Plaintext Seed RC4 PRNG Key Stream CRC-32 Integrity Check Value (ICV) IV Cipher text Message **Figure 8-2—WEP encapsulation block diagram** *Id.* at PAONE0015514.

encrypting a random session object key in a block cipher encryption process with the at least one object key;	TKIP encrypts a random session object key in a block cipher encryption process with the object key. The random session object key is an object key; it includes data and methods that operate on the data. The random session object key's data includes a temporal key value (TK) and a TKIP sequence counter value (TSC). The temporal key is a pseudorandom key value used in the process of encrypting MPDUs. It is generated by a pseudorandom function based on seeding from a master key and one or more random values called "nonces." *See, e.g.*, *id.* at PAONE0015552-553. The TSC is a number used to count the times the temporal key has been used to encrypt a block of plaintext data, i.e., the number of blocks of data encrypted with the temporal key. *See, e.g.*, *id.* at PAONE0015533. The TSC is set to 1 when a new temporal key value is installed. *See, e.g.*, *id.* at PAONE0015533. When the TSC reaches its maximum value, the temporal key expires and must be replaced through a re-keying process. *See, e.g.*, *id.* at PAONE0015521. The TSC is also used in encrypting MPDUs; without the dynamic TSC, the per-frame key would be the same value for each block of data. The methods that operate on the data include the function that increments the TSC value for each block of data that is encrypted using the temporal key and the pseudorandom function that calculates the temporal key value. The random session object key (TK and TSC) is encrypted in a block cipher encryption process. For example, TKIP is a block cipher encryption process, and TKIP's phase 2 key-mixing function uses a substitution box to encrypt the random session object key. *See, e.g.*, *id.* at PAONE0015532. The random session object key is encrypted in the phase 2 key-mixing function with the TTAK component of the object key. "TKIP encodes the TSC value from the sender to the receiver as a WEP IV and extended IV. . . . TKIP uses a cryptographic mixing function to combine a temporal key, the TA, and the TSC into the WEP seed. The receiver recovers the TSC from a received MPDU and utilizes the mixing function to compute the same WEP seed needed to correctly decrypt the MPDU. The key mixing function is designed to defeat weak-key attacks against the WEP key." *Id.* at PAONE0015520.

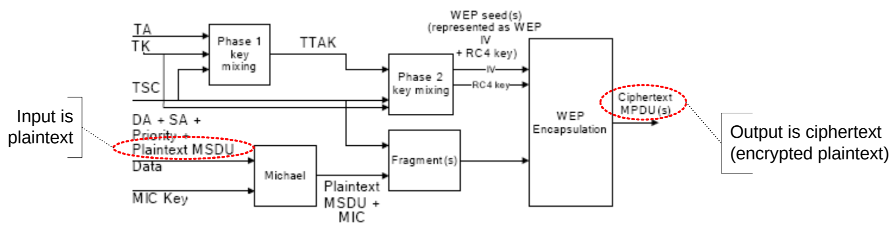
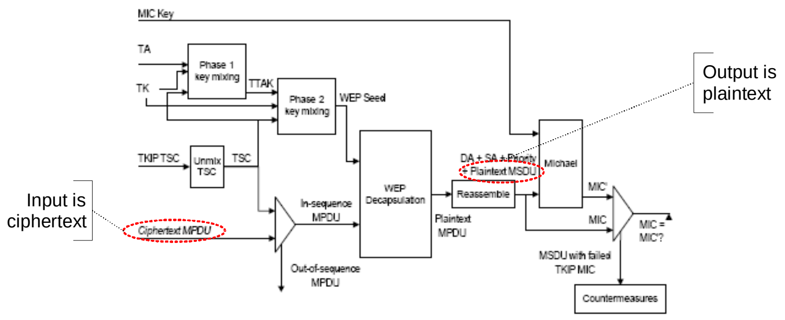
	 TA TK TSC Phase 1 key mixing TTAK WEP seed(s) (represented as WEP IV + RC4 key) Phase 2 key mixing IV RC4 key WEP Encapsulation Ciphertext MPDU(s) DA + SA + Priority + Plaintext MSDU Data Michael MIC Key Plaintext MSDU + MIC Fragment(s) **Figure 8-4—TKIP encapsulation block diagram** *Id.* at PAONE0015520.
encrypting a block of input plaintext data utilizing said key schedule;	TKIP encrypts blocks of input plaintext data utilizing a key schedule. For example, as illustrated in Figures 8-2 and 8-4 of the IEEE 802.11 Standard, blocks (MPDUs) of plaintext data are encrypted into ciphertext MPDUs using the key schedule. The creation of the key schedule values is discussed above. The key schedule values are XOR'd with the input plaintext data block to create a ciphertext block. Each per-frame key (a.k.a. WEP seed) is sent to the WEP encapsulation function, which treats the per-frame key as a WEP initialization vector and a WEP key The WEP initialization vector and WEP key are concatenated and thereafter are represented to the WEP encapsulation function as a WEP seed The pseudorandom number generator uses the per-frame key to create as many sub-keys as are needed to encrypt the plaintext block (MPDU) Another input into the WEP encapsulation function is the plaintext block (MPDU) to be encrypted The WEP encapsulation function prepares the block (MPDU) for encryption by performing a basic WEP integrity-checking procedure The plaintext block (MPDU) is mathematically combined with the sub-keys to produce a ciphertext sub-block, which can be safely communicated by radio Initialization Vector (IV) WEP Key Plaintext Seed RC4 PRNG Key Stream IV Cipher text CRC-32 Integrity Check Value (ICV) Message **Figure 8-2—WEP encapsulation block diagram** *Id.* at PAONE0015514.
modifying the at least one object key based on seeding from the random session object key;	TKIP modifies the object key based on seeding from the random session object key. For example, as illustrated in the IEEE 802.11 Standard Figure 8-4, the TKIP-mixed transmit address and key value (TTAK) is modified in the phase 2 key mixing function for each block of input data to generate a WEP seed. Specifically, the encryption process inputs the TKIP-mixed transmit address and key value (TTAK), the temporal key value (TK), and the TKIP sequence counter value (TSC) to the phase 2 key mixing function to generate the WEP

	seed. For example, as discussed above, the temporal key value (TK) and TKIP sequence counter value (TSC) are random session object key data. TKIP uses both the TK and TSC as seeds to modify the object key. Phase 1 key mixing uses the TK and TSC to modify the TTAK, the initial state of the object key. Phase 2 key mixing uses the TK and TSC to modify the per-frame key, the subsequent states of the object key, for each block of input plaintext data. *See, e.g.*, *id.* at PAONE0015530-531.  TA TK TSC Phase 1 key mixing TTAK WEP seed(s) (represented as WEP IV + RC4 key) Phase 2 key mixing IV RC4 key WEP Encapsulation Ciphertext MPDU(s) DA + SA + Priority + Plaintext MSDU Data MIC Key Michael Plaintext MSDU + MIC Fragment(s) **Figure 8-4—TKIP encapsulation block diagram** *Id.* at PAONE0015520.
modifying the key schedule based upon the at least one modified object key;	TKIP modifies the key schedule based upon the modified object key. The object key, the per-frame key, is modified for each block, then the new object key value is used to modify the key schedule values. *See, e.g.*, *id.* at PAONE0015520-521, PAONE0015513-514.  Initialization Vector (IV) WEP Key Plaintext Seed RC4 PRNG Key Stream IV Cipher text CRC-32 Integrity Check Value (ICV) Message **Figure 8-2—WEP encapsulation block diagram** *Id.* at PAONE0015514.
encrypting a next block of input plaintext data utilizing said modified key schedule; and	TKIP encrypts a next block of input plaintext data utilizing the modified key schedule according to the same method described above for encrypting a block of input plaintext data utilizing said key schedule. For example, TKIP receives super-blocks of plaintext data (MSDUs) serially, fragments the MSDUs into smaller plaintext

	blocks if necessary, and encrypts the resulting plaintext blocks (MPDUs) into ciphertext. "[F]ragmentation: The process of segmenting a medium access control (MAC) service data unit (MSDU) . . . into a sequence of smaller MAC protocol data units (MPDUs) prior to transmission." *Id.* at PAONE0015362. TKIP encrypts the resulting sequence of blocks (MPDUs) serially until no more unencrypted blocks remain. As discussed above, the key schedule is modified for each MPDU, and the modified key schedule is used to encrypt the MPDU. "TKIP represents the WEP seed as a WEP IV and ARC4 key and passes these with each MPDU to WEP for . . . encryption of the plaintext MPDU . . . ." *Id.* at PAONE0015520.
repeating the steps of modifying the at least one object key, modifying the key schedule and encrypting utilizing the modified key schedule until the encrypting of blocks of plaintext data is completed.	TKIP repeats the steps of modifying the object key, modifying the key schedule, and encrypting utilizing the modified key schedule until the encrypting of blocks of plaintext data is completed, as illustrated in the IEEE 802.11 Standard Figures 8-2 and 8-4, and as described in the relevant portions of the claim chart above. "For each MPDU, TKIP uses the [phase 2] key mixing function to compute the WEP seed." *Id.* at PAONE0015520. The term "WEP seed" is synonymous with the per-frame key component of the object key. Thus, the phase 2 key-mixing function modifies the object key for each block. TKIP uses the modified values in the per-frame key component of the object key to modify the key schedule values, as discussed above, then uses the modified key schedule to encrypt the block plaintext data, also discussed above. Each step in this process repeats until no more unencrypted blocks (MPDUs) remain.
2. A computer implemented method as defined in claim 1, wherein the modification of the key schedule is independent of the input plaintext data.	TKIP modifies the key schedule, as described above with respect to claim 1, in a way that is independent of the input plaintext data. As described above with respect to claim 1, the key schedule is derived by expanding the object key. The object key includes an initial data value (TTAK) and subsequent values (per-frame keys) generated by the phase 2 key-mixing function. Both the initial value and subsequent values are independent of the input plaintext data, as illustrated in Figures 8-2 and 8-4 of the IEEE 802.11 Standard.

	 TA TK TSC Phase 1 key mixing TTAK WEP seed(s) (represented as WEP IV + RC4 key) IV RC4 key Phase 2 key mixing WEP Encapsulation Ciphertext MPDU(s) DA + SA + Priority + Plaintext MSDU Data Michael MIC Key Plaintext MSDU + MIC Fragment(s) **Figure 8-4—TKIP encapsulation block diagram** *Id.* at PAONE0015514. As shown in the above diagram, the object key values are created based on the input values TA, TK, and TSC (transmit address, temporal key, and TKIP sequence counter). None of these input values depends on the input plaintext data. Thus the object key is independent of the plaintext data. Because the object key is always independent of the input plaintext data, and because the key schedule is created based on the object key, modification of the key schedule is independent of the input plaintext data.
23. A cryptographic communications system comprising:	Devices communicating using TKIP Technology constitute a cryptographic communications system. IEEE 802.11 is a standard that defines communications systems. Specifically, IEEE 802.11 defines wireless network communications systems: "The scope of this standard is to define . . . specifications for wireless connectivity for fixed, portable, and moving stations (STAs) within a local area." IEEE 802.11 Standard at PAONE0015355. IEEE 802.11 defines several cryptographic layers, including TKIP. "[T]his standard . . . [d]escribes the requirements and procedures to provide data confidentiality of user information being transferred over the wireless medium (WM) and authentication of IEEE 802.11-conformant devices." *Id.* at PAONE0015355, PAONE0015519.
at least two networked computer systems linked by a communication channel; and	As described above, IEEE 802.11 defines networks of computer systems linked by communication channels. IEEE 802.11 in general - and TKIP in particular - operate on computer systems. The IEEE 802.11 standard "provides a C-language reference implementation of the temporal key mixing function" at the center of the TKIP process. *Id.* at PAONE0016461. This C-language reference implementation is sample source code that can be compiled into executable machine code for use by a

	computer system. IEEE 802.11 defines several communication channels for linking computer systems, including, for example, radio communication channels. IEEE 802.11 defines "channel" as "[a]n instance of communications medium use for the purpose of passing protocol data units (PDUs) between two or more stations (STAs)" and defines "channel spacing" as "[t]he difference between the center frequencies of two nonoverlapping and adjacent channels of the radio transmitter." *Id.* at PAONE0015360.
each computer system including a central processing unit and a memory storage device for executing a block cipher encryption/ decryption process;	Each computer system has a central processing unit and a memory storage device for executing a block cipher encryption and decryption process. TKIP is a block cipher encryption/decryption process that is implemented on computer systems that include a central processing unit and a memory storage device. *See, e.g., id.* at PAONE0016461-485 (computer source code implementation of TKIP functions). TKIP is a block cipher encryption/decryption process because it operates by encrypting or decrypting blocks of data at a time. *See, e.g., id.* at PAONE0015514, PAONE0015520. For example, TKIP operates on blocks of data called medium access control protocol data units (MPDUs). *See, e.g., id.* at PAONE0015364. TKIP is implemented on computer systems having a processing unit and a memory storage device. This is demonstrated throughout the IEEE 802.11 standard. For example, with respect to the phase 2 key-mixing function, a component of TKIP, the 802.11 standard notes that "[t]he rotate and addition operations in STEP2 make Phase 2 particularly sensitive to the endian architecture of the processor . . . ." *Id.* at PAONE0015532. The 802.11 standard also makes clear that units of memory such as "bits," "bytes," and "ints" are used to store computed values throughout TKIP. *See, e.g., id.* at PAONE0016466-467.
wherein the encryption process transforms an input plaintext message to a ciphertext message and the decryption process transforms the ciphertext message to the input plaintext message, the encryption/	TKIP provides an encryption process that transforms an input plaintext message to a ciphertext message, and a decryption process that transforms the ciphertext message to the input plaintext message. For example, as illustrated in the IEEE 802.11 Standard Figure 8-4, blocks (MPDUs) of plaintext data are encrypted into ciphertext MPDUs, and as illustrated in Figure 8-5, ciphertext MPDUs are decrypted into plaintext MPDUs.

decryption process using at least one dynamic object key which is modified using a non-linear function for each block of input data, each object key being associated with a different key schedule to encrypt/decrypt the input plaintext/output ciphertext message.	 TA TK Phase 1 key mixing TTAK WEP seed(s) (represented as WEP IV + RC4 key) IV RC4 key TSC Phase 2 key mixing WEP Encapsulation Ciphertext MPDU(s) Input is plaintext Output is ciphertext (encrypted plaintext) DA + SA + Priority + Plaintext MSDU Data Michael Fragment(s) MIC Key Plaintext MSDU + MIC Figure 8-4—TKIP encapsulation block diagram IEEE 802.11 Standard at PAONE0015520.  MIC Key TA Phase 1 key mixing TK TTAK Phase 2 key mixing WEP Seed Output is plaintext TKIP TSC Unmix TSC TSC WEP Decapsulation DA + SA + Priority + Plaintext MSDU Michael Input is ciphertext In-sequence MPDU Reassemble MIC' MIC Ciphertext MPDU Plaintext MPDU MIC = MIC? Out-of-sequence MPDU MSDU with failed TKIP MIC Countermeasures Figure 8-5—TKIP decapsulation block diagram *Id.* at PAONE0015521. TKIP's encryption and decryption processes use at least one dynamic object key which is modified using a non-linear function for each block of input data. For example, as illustrated in the IEEE 802.11 Standard Figures 8-4 and 8-5, the TKIP-mixed transmit address and key value (TTAK) is modified using a non-linear function for each block of input data and/or ciphertext to generate a WEP seed. The TKIP-mixed transmit address and key (TTAK) is the initial state of the dynamic object key. In subsequent iterations of the encryption or decryption process, the state of the dynamic object key data is the per-frame key, also called a WEP seed. *See, e.g., id.* at PAONE0015513, PAONE0015520. The per-frame key is modified using a non-linear function for each block of input data. The per-frame key is so named because it is modified for each frame or block of data. *See, e.g., id.* at PAONE0015366. The phase 2 key-mixing function modifies the per-frame key. The phase 2 key-mixing function is a non-linear function because it employs a non-linear operation called a substitution box or "S-box." Both Phase 1 and Phase 2 rely on an S-box, defined

in this subclause. The S-box substitutes one 16-bit value with another 16-bit value. This function may be implemented as a table look up.

NOTE—The S-box is a nonlinear substitution.

Id. at PAONE0015529.

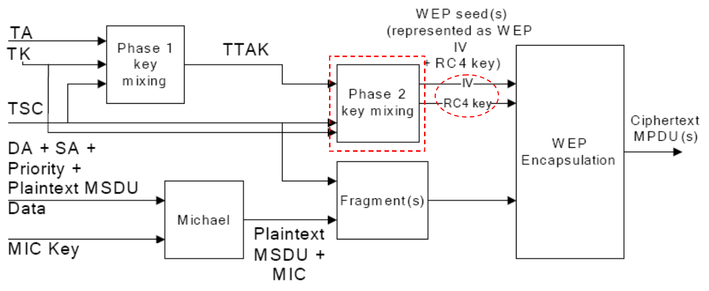


Figure 8-4—TKIP encapsulation block diagram

Id. at PAONE0015520.

As shown in the IEEE 802.11 Standard Figure 8-5, the TKIP *de*cryption process also uses the same at least one dynamic object key, the per-frame key, which is modified using the same non-linear function, the phase 2 key-mixing function, for each block of input data or MPDU to be decrypted.

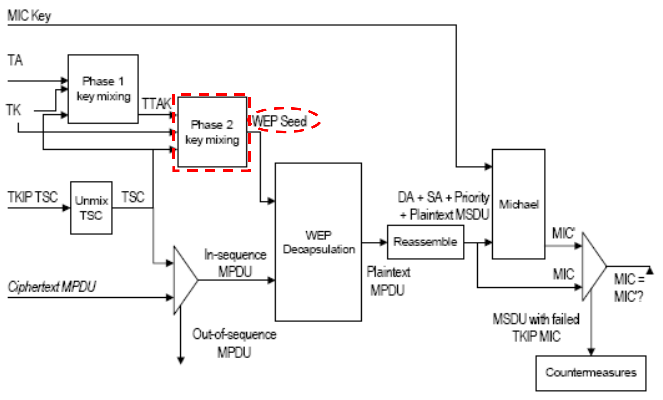
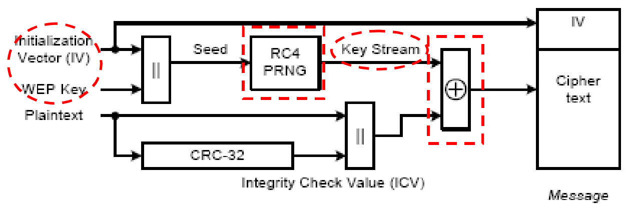
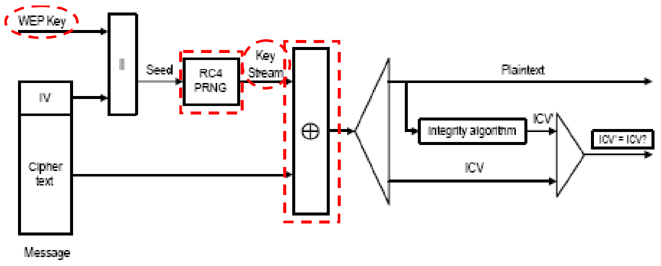
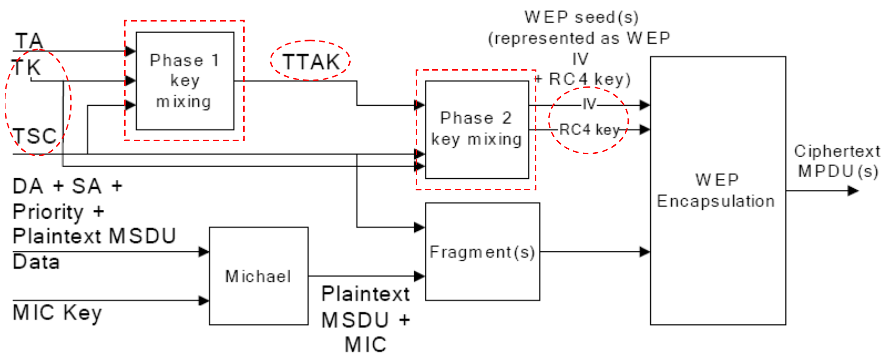


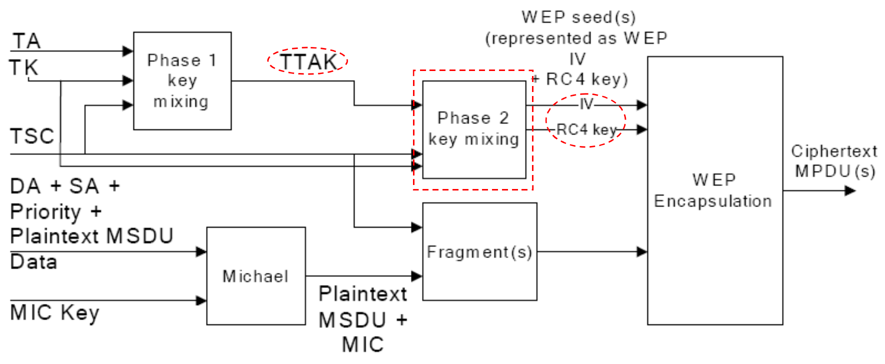
Figure 8-5—TKIP decapsulation block diagram

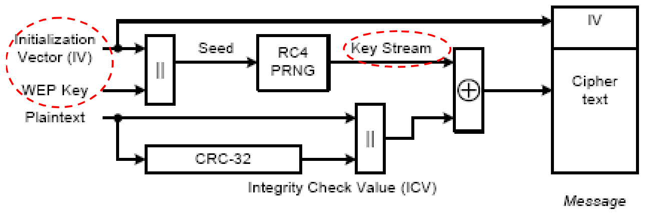
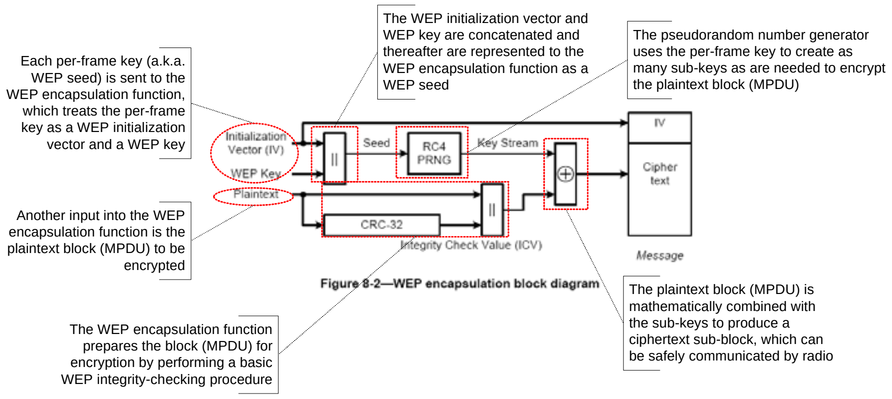
TKIP uses each object key value to generate a different key schedule, then uses the key schedule to encrypt each input plaintext message or decrypt the output ciphertext message. As illustrated in the IEEE 802.11 Standard Figures 8-2 and 8-3, each per frame key is associated with a different key schedule to encrypt the input plaintext and decrypt the output ciphertext message.

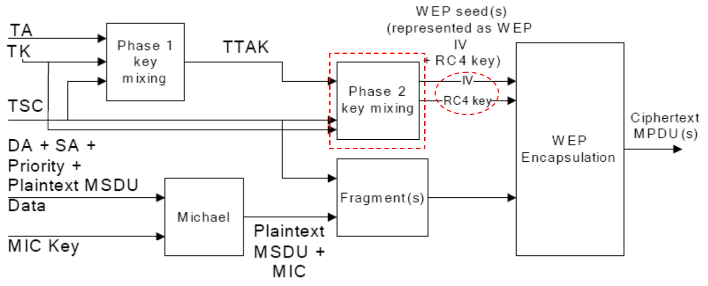
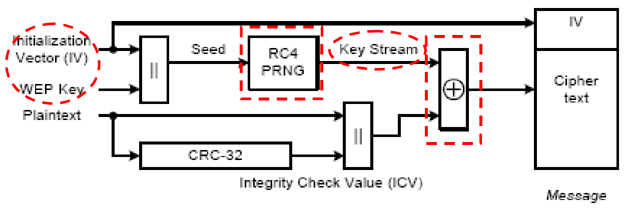
The object key is passed to the WEP encapsulation function as a per-frame key or WEP seed, which is composed of an initialization

	vector and a WEP key. The per-frame key is used as an input to the RC4 pseudorandom number generator function (PRNG), which expands the 128-bit per-frame key value into a set of keys long enough to encrypt the input plaintext message. *See, e.g.*, *id.* at PAONE0015513-514, PAONE0015531. An MPDU may be as long as 2,346 bytes, and TKIP requires a key schedule of equal length, so TKIP expands the per-frame key's 128 bits of data to create a key schedule as long as 18,768 bits (2,346 bytes). *See, e.g.*, *id.* at PAONE0015513. The key schedule is shown as the "key stream" in the diagrams below.  Initialization Vector (IV) WEP Key Plaintext Seed RC4 PRNG Key Stream CRC-32 Integrity Check Value (ICV) IV Cipher text Message Figure 8-2—WEP encapsulation block diagram *Id.* at PAONE0015514. TKIP uses the same process to generate a key schedule which is used to *de*crypt input ciphertext messages.  WEP Key IV Cipher text Message Seed RC4 PRNG Key Stream Plaintext Integrity algorithm ICV' ICV ICV' = ICV? Figure 8-3—WEP decapsulation block diagram *Id.*
24. A cryptographic communications system as defined in claim 23, wherein the encryption/ decryption process further includes the use of a random session object key having an initial state randomly generated by the computer system, and wherein the object key	The TKIP encryption/decryption process described above with respect to claim 23 uses a random session object key that has an initial state randomly generated by the computer system, and the object key modifications are based on seeding from the random session object key. The random session object key includes data and methods that operate on the data. The random session object key's data includes a temporal key value (TK) and a TKIP sequence counter value (TSC). The temporal key is a pseudorandom key value used in the process of encrypting and decrypting MPDUs. Its initial state is generated by a pseudorandom function based on seeding from a master key and one or more random values called "nonces." *See, e.g.*, *id.* at PAONE0015552-553. The TSC is a number used to count the times the temporal key has been used to encrypt a block

modifications are based on seeding from the random session object key.	of plaintext data, i.e., the number of blocks of data encrypted with the temporal key. *See, e.g.*, *id.* at PAONE0015533. The methods that operate on the data include the function that increments the TSC value for each block of data that is encrypted using the temporal key and the pseudorandom function that calculates the temporal key value. Object key modifications in TKIP are based on the random session object key, the TSC. TKIP modifies the object key based on seeding from the random session object key. As discussed above, the temporal key value (TK) and TKIP sequence counter value (TSC) are the random session object key data. TKIP uses both the TK and TSC as seeds to modify the object key. Phase 1 key mixing uses the TK and TSC to modify the TTAK, the initial state of the object key. Phase 2 key mixing uses the TK and TSC to modify the per-frame key, the subsequent states of the object key. *See, e.g., id.* at PAONE0015530-531.  TA TK TSC Phase 1 key mixing TTAK WEP seed(s) (represented as WEP IV + RC4 key) IV RC4 key Phase 2 key mixing DA + SA + Priority + Plaintext MSDU Data MIC Key Michael Plaintext MSDU + MIC Fragment(s) WEP Encapsulation Ciphertext MPDU(s) **Figure 8-4—TKIP encapsulation block diagram** *Id.* at PAONE0015520.
32. A computer implemented method for encrypting data comprising the steps of:	TKIP provides a computer implemented method for encrypting data. *See, e.g.*, IEEE Std 802.11-2007, at PAONE0015307-16538 (“IEEE 802.11 Standard”); IEEE Std 802.11i-2004, at PAONE0004198-4387 (“IEEE 802.11i Standard”); Wi-Fi Protected Access (WPA), at PAONE0004828-4882 (“WPA Specification”). TKIP was designed to provide data encryption superior to WEP, an earlier encryption standard, without requiring substantially more computing power than WEP required. *See, e.g.*, IEEE 802.11 Standard at PAONE0015392 (“The TKIP is a cipher suite enhancing the WEP protocol on pre-RSNA hardware.”). TKIP is and has been implemented on a computer. *See, e.g., id.* at PAONE0016461-485 (computer source code implementation of TKIP functions).
creating at least one	TKIP creates an object key in a block cipher that comprises data

object key in a block cipher, the at least one object key comprising data and methods that operate on said data;	and methods that operate on the data. TKIP is a block cipher because it operates by encrypting or decrypting blocks called medium access control protocol data units (MPDUs). *See, e.g.*, *id.* at PAONE0015364, PAONE0015520. TKIP creates an object key comprising data and methods that operate on said data. The initial state of the object key is the TKIP-mixed transmit address and key (TTAK). The initial state of the data includes the "keying material in[] the 80-bit TTAK." *Id.* at PAONE0015531. The TKIP object key also comprises methods that operate on said data, including the phase 2 key-mixing function. The phase 2 key-mixing function operates on the data in the TTAK to produce subsequent modified versions of the object key, described as per-frame keys, for each block of input plaintext data. *See, e.g.*, *id.* at PAONE0015520-521, PAONE0015529, PAONE0015531. Thus, in subsequent iterations of the process, the state of the object key data is the per-frame key, also called a WEP seed. The 128-bit per-frame key consists of a 24-bit WEP initialization vector (IV) and a 104-bit RC4 key. *See, e.g.*, *id.* at PAONE0015513, PAONE0015520.  TA TK TSC Phase 1 key mixing TTAK WEP seed(s) (represented as WEP IV + RC4 key) Phase 2 key mixing IV RC4 key WEP Encapsulation Ciphertext MPDU(s) DA + SA + Priority + Plaintext MSDU Data MIC Key Michael Plaintext MSDU + MIC Fragment(s) **Figure 8-4—TKIP encapsulation block diagram** *Id.* at PAONE0015520. For example, as illustrated above in Figure 8-4 of the IEEE 802.11 Standard, the block cipher encryption process inputs the transmitter address value (TA), the temporal key value (TK), and the TKIP sequence counter value (TSC) to the phase 1 key mixing function to create a TKIP-mixed transmit address and key value (TTAK). The block cipher encryption process also inputs the TKIP-mixed transmit address and key value (TTAK), the temporal key value (TK), and the TKIP sequence counter value (TSC) to the phase 2 key mixing function to generate the WEP seed. *See also, e.g.*, *id.* at

	PAONE0015519-533.
creating a key schedule based upon the at least one object key;	TKIP creates a key schedule based on the object key. For example, TKIP uses the per-frame key to create a key schedule for each block to be encrypted. The per-frame key's 128-bit key value is expanded, using the RC4 pseudorandom number generator, to the length necessary to encrypt an entire block of data. *See, e.g.*, *id.* at PAONE0015513-514, PAONE0015531. An MPDU may be as long as 2,346 bytes, and TKIP requires a key schedule of equal length. *See, e.g.*, *id.* at PAONE0015513. Accordingly, TKIP expands the per-frame key's 128 bits of data to create a key schedule as long as 18,768 bits (2,346 bytes).  Initialization Vector (IV) WEP Key Plaintext Seed RC4 PRNG Key Stream IV Cipher text CRC-32 Integrity Check Value (ICV) Message Figure 8-2—WEP encapsulation block diagram *Id.* at PAONE0015514.
encrypting a block of input plaintext data utilizing said key schedule;	TKIP encrypts blocks of input plaintext data utilizing a key schedule. For example, as illustrated in Figures 8-2 and 8-4 of the IEEE 802.11 Standard, blocks (MPDUs) of plaintext data are encrypted into ciphertext MPDUs using the key schedule. The creation of the key schedule values is discussed above. The key schedule values are XOR'd with the input plaintext data block to create a ciphertext block.  Each per-frame key (a.k.a. WEP seed) is sent to the WEP encapsulation function, which treats the per-frame key as a WEP initialization vector and a WEP key The WEP initialization vector and WEP key are concatenated and thereafter are represented to the WEP encapsulation function as a WEP seed The pseudorandom number generator uses the per-frame key to create as many sub-keys as are needed to encrypt the plaintext block (MPDU) Another input into the WEP encapsulation function is the plaintext block (MPDU) to be encrypted The WEP encapsulation function prepares the block (MPDU) for encryption by performing a basic WEP integrity-checking procedure The plaintext block (MPDU) is mathematically combined with the sub-keys to produce a ciphertext sub-block, which can be safely communicated by radio Initialization Vector (IV) WEP Key Plaintext Seed RC4 PRNG Key Stream IV Cipher text CRC-32 Integrity Check Value (ICV) Message Figure 8-2—WEP encapsulation block diagram *Id.* at PAONE0015514.
modifying the at least one object key using at least a non-linear	TKIP modifies the object key using a non-linear function. The phase 2 key-mixing function modifies the object key. The phase 2 key-mixing function is a non-linear function because it employs a

function;	non-linear operation called a substitution box or "S-box." The IEEE 802.11 Standard makes this explicitly clear. Both Phase 1 and Phase 2 rely on an S-box, defined in this subclause. The S-box substitutes one 16-bit value with another 16-bit value. This function may be implemented as a table look up. NOTE—The S-box is a nonlinear substitution. *Id.* at PAONE0015529.  TA TK TSC Phase 1 key mixing TTAK WEP seed(s) (represented as WEP IV + RC4 key) IV RC4 key Phase 2 key mixing WEP Encapsulation Ciphertext MPDU(s) DA + SA + Priority + Plaintext MSDU Data MIC Key Michael Plaintext MSDU + MIC Fragment(s) **Figure 8-4—TKIP encapsulation block diagram** *Id.* at PAONE0015520.
modifying the key schedule based upon the at least one modified object key;	TKIP modifies the key schedule based upon the modified object key. The object key, the per-frame key, is modified for each block, then the new object key value is used to modify the key schedule values. *See, e.g., id.* at PAONE0015520-521, PAONE0015513-514.  Initialization Vector (IV) WEP Key Plaintext Seed RC4 PRNG Key Stream CRC-32 Integrity Check Value (ICV) IV Cipher text Message **Figure 8-2—WEP encapsulation block diagram** *Id.* at PAONE0015514.
encrypting a next block of input plaintext data utilizing said modified key schedule; and	TKIP encrypts a next block of input plaintext data utilizing the modified key schedule according to the same method described above for encrypting a block of input plaintext data utilizing said key schedule. For example, TKIP receives super-blocks of plaintext data (MSDUs) serially, fragments the MSDUs into smaller plaintext blocks if necessary, and encrypts the resulting plaintext blocks

	(MPDUs) into ciphertext. "[F]ragmentation: The process of segmenting a medium access control (MAC) service data unit (MSDU) . . . into a sequence of smaller MAC protocol data units (MPDUs) prior to transmission." *Id.* at PAONE0015362. TKIP encrypts the resulting sequence of blocks (MPDUs) serially until no more unencrypted blocks remain. As discussed above, the key schedule is modified for each MPDU, and the modified key schedule is used to encrypt the MPDU. "TKIP represents the WEP seed as a WEP IV and ARC4 key and passes these with each MPDU to WEP for . . . encryption of the plaintext MPDU . . . ." *Id.* at PAONE0015520.
repeating the steps of modifying the at least one object key, modifying the key schedule and encrypting utilizing the modified key schedule until the encrypting of blocks of plaintext data is completed.	TKIP repeats the steps of modifying the object key, modifying the key schedule, and encrypting utilizing the modified key schedule until the encrypting of blocks of plaintext data is completed, as illustrated in the IEEE 802.11 Standard Figures 8-2 and 8-4, and as described in the relevant portions of the claim chart above. "For each MPDU, TKIP uses the [phase 2] key mixing function to compute the WEP seed." *Id.* at PAONE0015520. The term "WEP seed" is synonymous with the per-frame key component of the object key. Thus, the phase 2 key-mixing function modifies the object key for each block. TKIP uses the modified values in the per-frame key component of the object key to modify the key schedule values, as discussed above, then uses the modified key schedule to encrypt the block plaintext data, also discussed above. Each step in this process repeats until no more unencrypted blocks (MPDUs) remain.
33. A computer implemented method as defined in claim 32, wherein the nonlinear function is a hashing function.	As explained above with respect to claim 32, the phase 2 key-mixing function is a nonlinear function. It is also a hashing function because it takes an 80-bit TTAK value, a 128-bit TK (temporal key) value, and a 48-bit TSC value and returns a fixed-length 128-bit per-frame key value. *See, e.g.*, *id.* at PAONE0015531. Indeed, the IEEE 802.11 specification repeatedly refers to the key mixing-functions as hashing functions. *See, e.g.*, *id.* at PAONE0015522 ("TKIP Phase 1 key hashing"), PAONE0015529 ("Sbox for hash"), PAONE0015531 ("The TA is mixed into the temporal key in Phase 1 of the hash function.").

34. A cryptographic communications systems as defined in claim 23, wherein the non-linear function is a hashing function.	As explained above with respect to claim 23, the phase 2 key-mixing function is a nonlinear function. It is also a hashing function because it takes an 80-bit TTAK value, a 128-bit TK (temporal key) value, and a 48-bit TSC value and returns a fixed-length 128-bit per-frame key value. *See, e.g.*, *id.* at PAONE0015531. Indeed, the IEEE 802.11 specification repeatedly refers to the key mixing-functions as hashing functions. *See, e.g.*, *id.* at PAONE0015522 ("TKIP Phase 1 key hashing"), PAONE0015529 ("Sbox for hash"), PAONE0015531 ("The TA is mixed into the temporal key in Phase 1 of the hash function.").